EP34727 (9)

PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 17/30

A1

(11) International Publication Number: WO 97/49048

(43) International Publication Date: 24 December 1997 (24.12.97)

(21) International Application Number: PCT/US97/10191

(22) International Filing Date: 17 June 1997 (17.06.97)

(30) Priority Data:

08/664,565	17 June 1996 (17.06.96)	US
08/794,425	5 February 1997 (05.02.97)	US

(71) Applicant: IDD ENTERPRISES, L.P. [US/US]; Suite 1810, 2 World Trade Center, New York, NY 10048 (US).

(72) Inventor: LI, Yanhong; 191 Spruce Mill Lane, Scotch Plains, NJ 07076 (US).

(74) Agent: GERSTEIN, Robert, M.; Marshall, O'Toole, Gerstein, Murray & Borun, 6300 Sears Tower, 233 S. Wacker Drive, Chicago, IL 60606-6402 (US).

(81) Designated States: AL, AM, AT, AU, AZ, BB, BG, BR, BY, CA, CH, CN, CZ, DE, DK, EE, ES, FI, GB, GE, HU, IL, IS, JP, KE, KG, KP, KR, KZ, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, TJ, TM, TR, TT, UA, UG, UZ, VN, ARIPO patent (GH, KE, LS, MW, SD, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

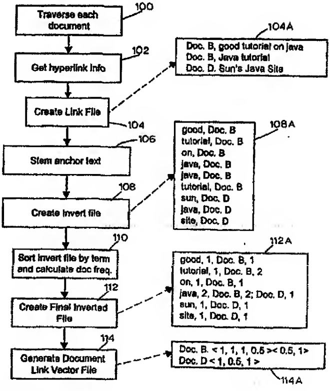
Published

With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: HYPERTEXT DOCUMENT RETRIEVAL SYSTEM AND METHOD

(57) Abstract

A search engine for retrieving documents pertinent to a query indexes documents in accordance with hyperlinks pointing to those documents. The indexer traverses the hypertext database and finds hypertext information including the address of the document the hyperlinks point to and the anchor text of each hyperlink. The information is stored in an inverted index file, which may also be used to calculate document link vectors for each hyperlink pointing to a particular document. When a query is entered, the search engine finds all document vectors for documents having the query terms in their anchor text. A query vector is also calculated, and the dot product of the query vector and each document link vector is calculated. The dot products relating to a particular document are summed to determine the relevance ranking for each document.



HYPERTEXT DOCUMENT RETRIEVAL SYSTEM AND METHOD

FIELD OF INVENTION

The present invention relates to hypertext document retrieval, and more particularly to systems and methods of searching databases distributed over wide-area networks such as the World Wide Web.

BACKGROUND OF THE ART

A hypertext is a database system which provides a unique and non-sequential method of accessing information using nodes and links. Nodes, i.e. documents or files, contain text, graphics, audio, video, animation, images, etc. while links connect the nodes or documents to other nodes or documents. The most popular hypertext or hypermedia system is the World Wide Web, which links various nodes or documents together using hyperlinks, thereby allowing the non-linear organization of text on the web.

A hyperlink is a relationship between two anchors, called the head and the tail of the hyperlink. The head anchor is the destination node or document and the tail anchor is the document or node from which the link begins. On the web, hyperlinks are generally identified by underscoring or highlighting certain text or graphics in a tail anchor

Search engines usually take a user query as input and attempt to find documents related to that query. Queries are usually in the form of several words which describe the subject matter of interest to the user. Most search engines operate by comparing the query to an index of a document collection in order to determine if the content of one or more of those documents matches the query. Since most casual users of search engines do not want to type in long, specific queries and tend to search on popular topics, there may be thousands of documents that are at least tangentially related to the query. When a search engine has indexed a large document collection, such as the Web, it is particularly likely that a very large number of documents will be found that have some relevance to the query. Most search engines, therefore, output a list of documents to the user where the documents are ranked by their degree of pertinence to the query and/or where documents having a relatively low pertinence are not identified to the user. Thus, the way in which a search engine determines the relevance ranking is extremely important in order to limit the number of documents a user must review to satisfy that user's information needs.

Almost all ranking techniques of search engines depend on the frequency of query terms in a given document. When other related factors are the same, the higher a term's frequency in a given document. the higher the relevance score of this document to a query including that term. Factors other than term frequency, such as such document frequency, i.e. how many documents contain the term, may also be taken

$$IDF_t = \frac{1}{DF_t}$$

Using an inverse document frequency insures that junk words such as "the," "of," "as," etc. do not have a high weight. In addition, when a query uses multiple terms, and one of those terms appears in many documents, using an IDF weighting gives a lower ranking to documents containing that term, and a higher ranking to document containing other terms in the query.

There are normalized versions of term weighting, which take into account the length of a document including a particular term. The assumption made is that the more frequently a term appears in a document for a given amount of text, the more likely that document is relevant to a query including that term. That assumption may not be true, however, in many cases. For example, if the query is "Java tutorial," a document (call it J), which contains 100 lines with each line consisting of just the phrase "Java tutorial," would get a very high relevance score and would be output by a search engine as one of the most relevant documents to the user. That document, however, would be useless to the user since it provides no information about a "Java tutorial." What the user really needs is a good tutorial for the Java programming language such as found on Sun's Java tutorial site (http://Java.sun.com/tutorial). Unfortunately, the phrase "Java tutorial" does not occur 100 times on Sun's site, and therefore most search which are in a language other than the language of the query entered by the search engine user. Translation tools are a possible solution, but they may be difficult and expensive to build.

In addition, traditional search engines may be unable to identify non-textual material which is relevant to a query. For instance, a Web site containing pictures of Mozart or examples of Mozart's music may not be deemed relevant by a search engine when that search engine can only search for the word "Mozart" within the text of documents.

SUMMARY OF THE INVENTION

A method of indexing documents includes obtaining a list of hyperlinks pointing to each document, where each hyperlink includes one or more terms. Each document is indexed with the terms in the hyperlink pointing to that document. A number of hyperlinks, each containing a particular term, may point to a document. The number of hyperlinks containing that particular term pointing to the document is indexed with that document.

A particular term may appear in hyperlinks pointing to a number of documents, and the number of documents having the particular term in hyperlinks pointing to those documents is indexed with that term. Indexing may include creating a file listing each term, the number of documents having that term in hyperlinks pointing to those documents, a for each hyperlink pointing to a particular document to obtain a summed relevance score for that document.

The query may be represented by a query vector where the query vector contains a dimension for each term in the query. Each document may be represented by document link vectors for each hyperlink pointing to the document, where each document link vector contains a dimension for each term in the corresponding hyperlink pointing to that document. Comparing the words in the query to the words in the hyperlinks includes calculating the dot product of the query vector with the document link vector for that hyperlink. Summing the relevance ranking for each hyperlink pointing to a document includes summing the dot products obtained using the document link vectors for a particular document to obtain the summed relevance score for that document. The summed relevance scores may then be compared to obtain a ranking of documents.

The dimension for a term in a query vector may be related to the inverse of the number of documents having a respective hyperlink containing that term pointing to those documents. Similarly, the dimension for a term in a document link vector may be related to the inverse of a number of documents having a respective hyperlink containing that term pointing to those documents.

Other features and advantages are inherent in the hypertext document retrieval system and method claimed and disclosed or will service provider or through any one or more of the other servers. Servers 13, 14, 15, and 16 include files of documents 17, 18, 19, and 20, respectively. Files 17, 18, 19, and 20 contain documents available to users of the network. Server 12 includes an index file 21 as discussed in more detail below. The server computer 12 traverses the network looking for all hypertext documents residing in the files 17-20 of the other server computers 13-16 in order to build the index file 21.

Fig. 2 describes the general structure of an indexing and retrieval system 30 of the present invention. A user from outside the system 30 inputs a query 32 through a user interface 34, which will typically reside on the user's computer, such as a client computer 10 (Fig. 1). The user's query is then transmitted through the network to the indexing and retrieval system 30, which generally resides on a server, such as server 12 (Fig. 1). The system 30 includes a retrieval engine 36, index files 38, and an index engine 40. The operation of the retrieval engine 36 and index engine 40 and the creation of the index files 38 are described below. The index engine 40 creates the index files 38 by traversing a document database 42, such as that found on the World Wide Web. The document database 42 might include files 17-20 (Fig. 1). The index files 38 created by the index engine 40 may take various forms in accordance with the present invention, but may include a link file 44, an inverted file 46, and a document vector file 48, all of which are described in detail below. The retrieval engine 36 uses the index files 38 in order to identifies the anchor text of the hyperlink 50. By identifying the phrase "good tutorial on Java" as the anchor text in the command 66, that phrase is thereby underlined in the text 56 of Document A. When text such as anchor text 64 is underlined, it alerts a reader of Document A to the existence of the hyperlink. When a user then clicks on the anchor text 64, the command 66 points to Document B, thereby instructing the user's computer to send a message to the address URL2, requesting a copy of Document B.

The author of Document A must, of course, create the command 66 and identify the anchor text 64. Generally, authors of such documents will describe, in that author's opinion, the head anchor document (in this case Document B) with the words of the anchor text (in this case, anchor text 64). Therefore, if there are many authors like the author of Document A that make link commands to document B using the anchor text 64, then a user looking for a Java tutorial is highly likely to be interested in the information in Document B.

Fig. 4 is a representation of a simple hypertext system having only four documents, Documents A, B, C, and D. The system shown in Fig. 4 has only three hyperlinks, hyperlink 50, also shown in Fig. 3, and hyperlinks 68 and 70. The anchor text "good tutorial on Java" in Document A is the tail for the hyperlink from Document A to Document B, as shown in Fig. 3. Document C contains two sets of anchor text "Java tutorial" and "Sun's Java site." The anchor text 72 in Document C points system may also collect a variety of information about the document including its title and possibly the text of the document. The system may also create an abstract, if desired.

At block 104, the system creates one or more link files where entries in the files have a format:

< doc.ID, anchor-text >

where doc.ID is an identifier for each head document of a hyperlink having the corresponding anchor text. The doc.ID may be in the form of a URL or may be another identifier which is indexed in some manner with the document's URL. Box 104A is an example of a link file, as referred to in Fig. 2, created for the database of the documents shown in Fig. 4. Since the database in Fig. 4 has three hyperlinks, there are three entries in file 104A. The system may also store the number of times a term appears in anchor text for a particular link. In the examples shown, each term only appears once in a particular link.

Although Fig. 5 shows that traversing of documents in block 100 occurs before link files are created at block 104, it is possible for some link files to be created prior finishing traversing all documents in the database. In fact, once the database has been entirely traversed, it may be desirable to update the link files and other index files by retraversing documents in order to determine if any additional documents have been added to the database, or if any hyperlinks have been added to the documents.

Control next passes to block 112, which creates final invert file as shown in 112A. Entries in the final invert file are in the format:

< term, DF, doc1, lf1, doc2, lf2, ..., doci, LFi >

where "term" is a term in the anchor text, DF is the document frequency for that term, doci is the document identifier for Document i, and LFi is the link term frequency for doci. Link term frequency is defined as the number of hyperlinks pointing to doci whose anchor text consists of the particular term. For example, the term "good" appears in only one hyperlink that points to Document B, so the link term frequency of the term "good" for Document B is one. The term "Java" appears in two hyperlinks that point to Document B, so the link term frequency of "Java" for Document B is two. One embodiment of the retrieval engine of the present invention will depend on this file to find documents related to a user query.

The index engine at box 114 may also generate a document link vector file where entries in the document link vector file are in the format of:

$$doc.id, v_1, v_2, \ldots, v_i$$

where doc.id is the identifier for a particular document, and v_i is a vector representation of a hyperlink found in the link file. Each vector v_i will be in the format of:

$$< w(t_1), w(t_2), \ldots, w(t_i) >$$

Thus, the only entries in the link vector files which need to be created are those pertaining to documents having query terms in the anchor text of hyperlinks pointing to those documents.

In the first vector for Document B, the first three dimensions are "one" since the terms "good," "tutorial," and "on" only appear in anchor text pointing to one document, and they only appear once in the anchor text. Thus:

$$TF*IDF = 1*1=1.$$

The term "Java," however, has a term frequency of one and document frequency of two, and therefore has an inverse document frequency of .5. Thus, TF*IDF for "Java" is .5, making the last dimension in the first vector for Document B equal to .5. The remaining dimensions in the second vector for Document B and the vector for Document D are also calculated according to the TF*IDF formula.

The link file 104A, the invert file 108A, the final invert file 112A, and the document link vector file 114 are all considered index files as shown in Fig. 2. Although the files as shown in Fig. 5 are preferred, there are many indexing techniques which can be used with a system of the present invention, which rely on anchor text and link frequency in order to index documents. For instance, the files may be compressed or have a variety of relational structures for the data within files or between files.

Referring now to Fig. 6, the retrieval process achieves relevance ranking by using the vector space model and link vector voting.

"tutorial" in the query. The IDF as previously calculated in box 110 of Fig. 5 for "Java" is .5 and as calculated for "tutorial" is one.

Once the query vector and all relevant document link vectors have been found or calculated, control passes to block 130 to calculate the relevance scores for each document. The relevance score is calculated by finding the dot product of each document link vector with the query vector. A dot product for vectors < a, b, c > and < d, e, f > is defined as:

$$\frac{a*d + b*e + c*f}{\sqrt{a^2+b^2+c^2}\sqrt{d^2+e^2+f^2}}$$

If two vectors do not have the same dimensions, a zero is entered for each dimension which is not present in that vector. For instance, the first vector for Document B is represented as:

< 1, 1, 1, 0.5 >.

In such an instance, the query vector would be represented as:

< 0, 1, 0, .5 >

so that the dimensions representing "tutorial" in each vector and "Java" in each vector match up. The dot product of the query vector with the first document link vector for Document B would then be calculated as follows:

$$\frac{0 \text{ x } 1 + 1 \text{ x } 1 + 0 \text{ x } 1 + .5 \text{ x } .5}{\sqrt{1^2+1^2+1^2+.5^2}\sqrt{1^2+.5^2}} = .620$$

A similar calculation for the second vector for Document B would lead to a dot product of 1.

combination with the hyperlinked based index and retrieval system of the present invention. This combination might be used in the case of a link-based relevance score tie, or merely to supplement the link-based information. For instance, suppose the relevance scores for Document A and C are 0.6 and 0.8, respectively, based on conventional and relevance ranking. The final relevance ranking for the query utilizing the conventional ranking to break the tie of the link-based ranking would be Document B, Document D, Document C, and Document A.

Another reason to use combination ranking may be when there are too few hyperlinks (such as only one link) pointing to a document. In such a case, the relevance score based upon the one link may not be accurate, so a threshold can be set for the link-based relevance score. If the link-based relevance score is lower than the threshold, other means of relevance ranking may be used or combined with the link-based relevance score.

Because the index files of the present invention use only hyperlink information, relevance ranking does not depend on the words appearing in documents themselves, or, if used in combination with conventional relevance ranking do not depend solely on words appearing in the documents. Instead, the relevance ranking depends on descriptions of those documents in the anchor text of hyperlinks pointing to the documents. Documents such as Document J described above will not have a high describes itself. Thus, in the examples shown above, Sun's Java tutorial site will receive a high summed relevance rank even though the term "Java tutorial" appears only once in the document.

The ranking method based on hyperlinks pointing to a given document can be used to select the most popular documents in a specific field using the feature words or description of that field as the query to the system. By analyzing the link file described in the preferred embodiment, and comparing the different descriptions of hyperlinks pointing to the same document, a system can automatically construct a thesaurus or synonym tool.

The foregoing detailed description has been given for clearness of understanding only, and no unnecessary limitations should be understood therefrom, as modifications would be obvious to those skilled in the art.

3. The method of claim 2 wherein the indexing comprises creating a file listing:

each term;

the number of documents having that term in hyperlinks pointing to those documents;

a document identifier for each document having that term in hyperlinks pointing to that document; and

the number of hyperlinks containing that term pointing to each identified document.

4. The method of claim 1 wherein:

a particular term may appear in hyperlinks pointing to a number of documents; and

the number of documents having the particular term in hyperlinks pointing to those documents is indexed with a document identifier for each document having the particular term in a hyperlink pointing to that document.

5. The method of claim 4 wherein each document having a particular term in a hyperlink pointing to that document is indexed with an inverse of the number of documents having the particular term in hyperlinks pointing to those documents.

10. A method of ranking documents based on the document's relevance to a query, wherein the query comprises at least one term, and wherein hyperlinks contain terms and point to corresponding documents, the method comprising:

comparing the words in the query to the words in a hyperlink to obtain a relevance ranking for each hyperlink; and

summing the relevance rankings for each hyperlink pointing to a particular document to obtain a summed relevance score for that document.

11. The method of claim 10 wherein:

a number of hyperlinks, each containing a particular term, may point to a document; and

the number of hyperlinks containing the particular term pointing to the document is indexed with that document.

12. The method of claim 11 wherein:

a particular term may appear in hyperlinks pointing to a number of documents; and

the number of documents having a particular term in hyperlinks pointing to those documents is indexed with that term.

16. The method of claim 10 wherein:

a term may appear a number of times in a hyperlink pointing to a document; and

the number of times each term appears in a hyperlink is indexed with the document pointed to by the hyperlink.

17. The method of claim 10 wherein the terms are stemmed words.

18. The method of claim 10 wherein:

the query is represented by a query vector wherein the query vector contains a dimension for each term in the query; and

each document is represented by document link vectors for each hyperlink pointing to the document, wherein each document link vector contains a dimension for each term in the corresponding hyperlink pointing to that document.

19. The method of claim 18 wherein comparing the words in the query to the words in the hyperlink comprises calculating the dot product of the query vector with the document link vector for that hyperlink.

25. A computer-readable memory device comprising a set of instructions for performing the method of claim 1.

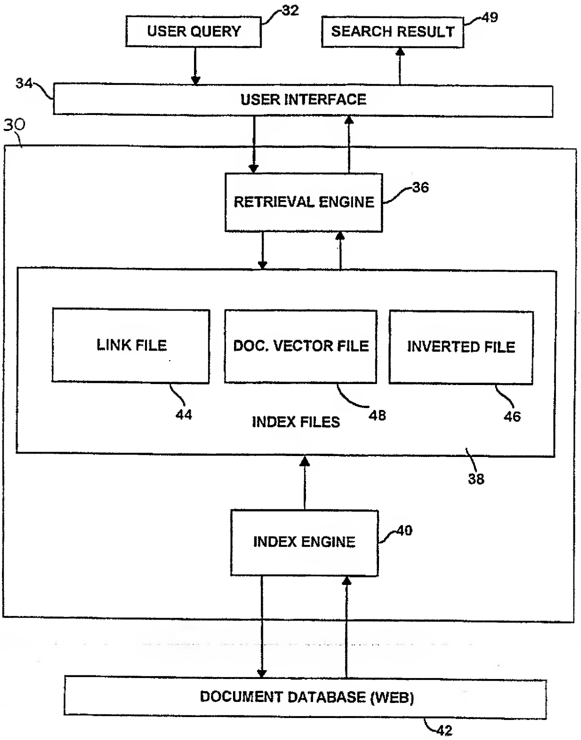

USER QUERY
32
SEARCH RESULT
49
34
USER INTERFACE
30
RETRIEVAL ENGINE
36
LINK FILE
DOC. VECTOR FILE
INVERTED FILE
44
INDEX FILES
48
46
38
INDEX ENGINE
40
DOCUMENT DATABASE (WEB)
42

FIG. 2

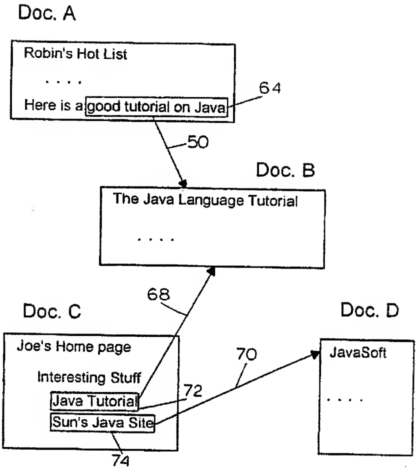
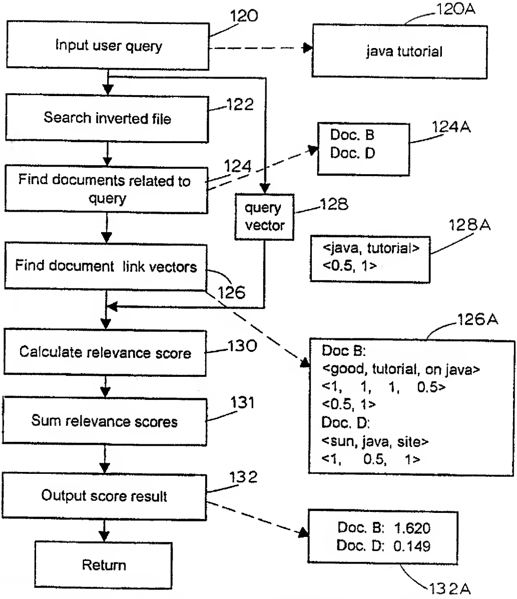

Doc. A
Robin's Hot List
. . . .
Here is a good tutorial on Java
64
50
Doc. B
The Java Language Tutorial
. . . .
68
Doc. C
Joe's Home page
Interesting Stuff
Java Tutorial
Sun's Java Site
72
74
70
Doc. D
JavaSoft
. . . .

FIG. 4

FIG. 6


120
Input user query
120A
java tutorial
122
Search inverted file
124A
Doc. B
Doc. D
124
Find documents related to query
query vector
128
128A
<java, tutorial>
<0.5, 1>
Find document link vectors
126
126A
Doc B:
<good, tutorial, on java>
<1, 1, 1, 0.5>
<0.5, 1>
Doc. D:
<sun, java, site>
<1, 0.5, 1>
Calculate relevance score
130
Sum relevance scores
131
Output score result
132
Doc. B: 1.620
Doc. D: 0.149
132A
Return

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 97/10191

C.(Continuation) DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication,where appropriate, of the relevant passages	Relevant to claim No.
A	DUNLOP M D ET AL: "Hypermedia and free text retrieval" INFORMATION PROCESSING & MANAGEMENT, 1993, UK, vol. 29, no. 3, ISSN 0306-4573, pages 287-298, XP002043306 see page 289, line 20 - page 290, line 23 ---	1-25
A	BICHTELER J ET AL: "The combined use of bibliographic coupling and cocitation for document retrieval" JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE, JULY 1980, USA, vol. 31, no. 4, ISSN 0002-8231, pages 278-282, XP002043307 see the whole document -----	1-25

1